# Chapter 2

# Case 1: Stock Returns

## Overview

In this project, you will investigate IBM and the aggregate stock market returns. You will compute the risk premium and the various risk measures for IBM and the aggregate stock market returns. You need to download the dataset "Case1CAPM.csv" on blackboard and program in R studio.

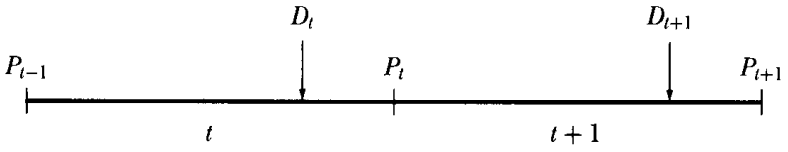
## Data Description

- DATE : Date formed as year month and day. For example, 3July2017 is entered as 20170703.
- IBMRET: **Daily** International Business Machines (IBM) stock returns **in percentage unit** defined in equation 2.1.
- MarketEXRET: **Daily market excess returns in percentage unit** defined as $R_{m,t} - r_{f,t}$. The excess returns on the market, value-weight returns of all CRSP firms incorporated in the US and listed on the NYSE, AMEX, or NASDAQ that have a CRSP share code of 10 or 11.
- RF: The Tbill return is the simple **daily rate (in percentage)** that over the number of trading days in the month compounds to 1-month TBill rate from Ibbotson and Associates Inc.
- Data Source: CRSP through WRDS and Fama French Data Library.

# 2.1 Stock Return

The rate of return an investor receives from buying a common stock and holding it for a given period of time is equal to the cash dividends received plus the capital gain (or minus the capital loss) during the holding period divided by the purchase price of the security. Denote by $D_t$ the asset's dividend payment at date $t$ and assume, purely as a matter of convention, that this dividend is paid just before the date-t price $P_t$ is recorded; hence $P_t$ is taken to be the ex-dividend price at date $t$. Alternatively, one might describe $P_t$ as an end-of-period asset price, as shown in the below figure. Then the net simple return at date $t$ may be defined as,

$$R_t = \frac{P_t + D_t}{P_{t-1}} - 1 \tag{2.1}$$




From this definition it is apparent that the asset's gross return over the most recent $k$ periods from date $t-k$ to date $t$, written $1 + R_t(k)$ is

$$R_t(k) = (1 + R_t)(1 + R_{t-1})...(1 + R_{t-k+1}) - 1 \equiv \prod_{j=0}^{k-1}(1 + R_{t-j}) - 1 \tag{2.2}$$

Among practitioners and in the financial press, a return-horizon of one year is usually assumed implicitly; hence, unless stated otherwise, a return of 20% is generally taken to mean an annual return of 20%. Moreover, multiyear returns are often annualized to make investments with different horizon comparable, thus,

$$Annualized\ R_t(k = h * 252) = \Big[\prod_{j=0}^{k-1}(1 + R_{t-j})\Big]^{1/h} - 1 \tag{2.3}$$

where h is the number of years and k is the number of days within these h years. For investments with daily $R_t$ horizon, practitioners compound the daily returns to obtain annualized returns. Here $k$ is the number of days in the past year,

$$Annualized\ R_t(252) = \Big[\prod_{j=0}^{252-1}(1 + R_{t-j})\Big] - 1 \tag{2.4}$$

Similarly, to compute the return of a five-year investment horizon from daily returns $R_t$, one should follow the equation below,

$$Annualized\ R_t(252 * 5) = \Big[ \prod_{j=0}^{252*5-1} (1 + R_{t-j}) \Big]^{1/5} - 1 \tag{2.5}$$

Here $k$ is the number of days in the past five years. Finally, to compute the return of a half-year investment horizon from daily returns $R_t$, one should follow the equation below,

$$Annualized\ R_t(252/2) = \Big[ \prod_{j=0}^{252/2-1} (1 + R_{t-j}) \Big]^{2} - 1 \tag{2.6}$$

Here $k$ is the number of days in the past five years.

## 2.2 Analysis

### 2.2.1 Summary Statistics

#### Step 0: Load Data

Read the variable description and convert the class of variable "DATE" into *Date*.

```
#Case 1: CAPM (Chap9)
#Author:Lai Xu
#Step 0.1: Prepare (data download, R script, WD)
rm(list = ls())
# remove all variables in the workspace

#Step 0.2: load the data
CAPM<-read.csv("Case1CAPM.csv", header = TRUE, sep=",")
#CAPM<-read.csv('Case1CAPM.csv', header = TRUE, sep=",")

#Step 0.3: Dimension and Names of the variables
dim(CAPM)
names(CAPM)
# 5540 rows (days), 4 variables; Time series data

#Step 0.4: Read data descriptions 2.3 + view Data
View(CAPM)

#Step 0.5: Change the class of variable DATE to be "Date"
class(CAPM$DATE) # integer
DATE<-as.Date(as.character(CAPM$DATE), "%Y%m%d")
```

**CAPM¡-read.csv("FileName", header=TRUE, sep=",")**

read.csv("FileName", header=TRUE, sep=",") reads a file in table format and creates a data frame from it. This data frame is named "CAPM". This dataset contains a header and it uses comma "," to separate each line of the file.

- "FileName" command will load the specific dataset in .csv format. Use straight quotes around the file name. Do not use curly quotes or spaces inside the blanks.
- header = TRUE is a logical value indicating whether the file contains the variable names as its first line.
- Sep = "," the field separator character. Values on each line of the file are separated by this character.

**as.Date(x, ...)** Functions to convert between character representations and objects of class "Date" representing calendar dates.

**as.character(x, ...)** Create objects of type "character".

```
## read in date info in format 'ddmmmyyyy'
x <- c("1jan1960", "2jan1960", "31mar1960", "30jul1960")
z <- as.Date(x, "%d%b%Y")

## read in date/time info in format 'm/d/y'
x <- c("02/27/92", "02/27/92", "01/14/92", "02/28/92", "02/01/92")
z <- as.Date(x, "%m/%d/%y")

#Code Value
#%d Day of the month (decimal number)
#%m Month (decimal number)
#%b Month (abbreviated)
#%B Month (full name)
#%y Year (2 digit)
#%Y Year (4 digit)
```

For more discussion on "Dates and Times in R", read the website `https://www.stat.berkeley.edu/~s133/dates.html`

## Step 1: Time-Series Plots of Stock Returns

### Step 1.1 Create the excess returns of IBM

```
##Step 1.1 Create the excess returns of IBM
ibmRET<-CAPM$IBMRET
marketEXERT<-CAPM$MarketEXRET
RF<-CAPM$RF
IBMEXERT<-ibmRET-RF
```

### Step 1.7 (FIG) Time-Series Plot of Daily Returns

Create two time-series plots. In both plots, x-axis is "DATE". In the first figure, y-axis is the market excess returns. In the second figure, y-axis is the excess returns of IBM.

```
## Step 1.3 Time-Series Plot of Yearly returns
jpeg(filename = "Case1_MRKEXERT.jpeg")
plot(DATE,marketEXERT,type = "l",xlab = "year",
ylab = "daily excess return(%)",
     main = "Market Excess Return (%)",
     , ylim = c(-15,15))
dev.off()
```

**jpeg(filename ="Case1_MRKEXERT.jpeg")** Graphics devices for JPEG format bitmap files.

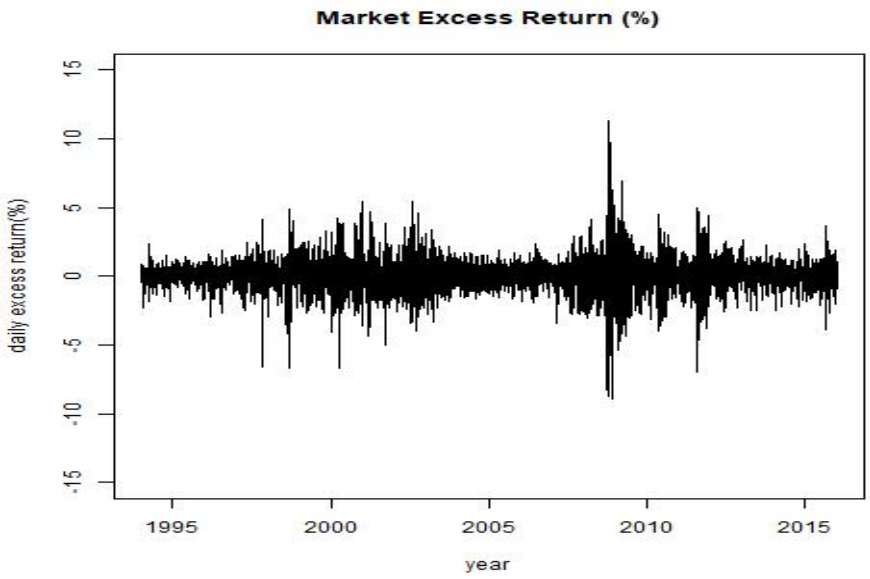
- **filename**: the name of the output file, up to 511 characters.

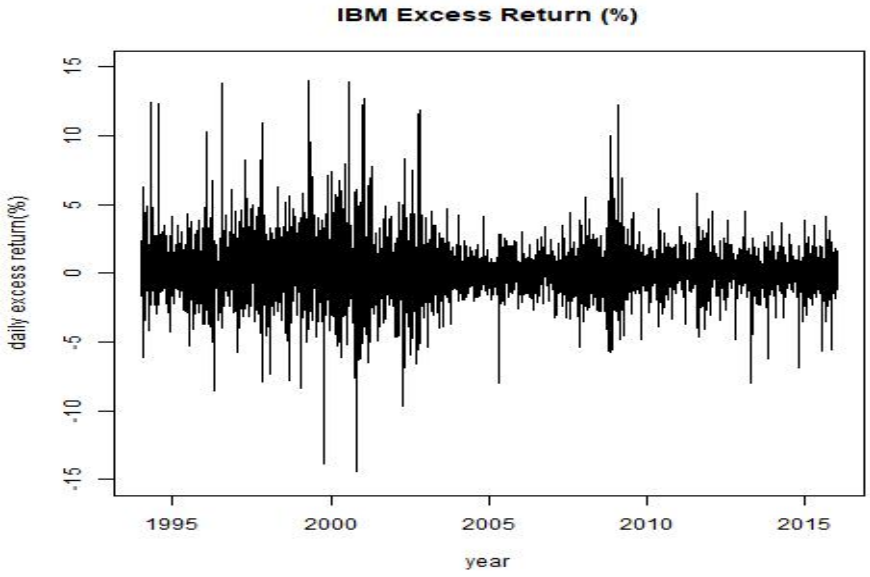
**dev.off()** ends the creation of the plot.

**plot(x,y,...)** is a generic function for plotting of R objects.

- **x**: the coordinates of points in the plot.
- **y**: the y coordinates of points in the plot, optional if x is an appropriate structure.
- **type**: what type of plot should be drawn.
- **col**: indicate the color of plot.
- **xlab**: a title for the x axis

- **ylab**: a title for the y axis
- **main**: an overall title for the plot
- **ylim**: numeric vector of length 2, giving the y coordinate range.
- **xlim**: numeric vector of length 2, giving the x coordinate range.



**Time-series plot**

The graph shows the daily market excess returns (percentage), from 1994-01-03 to 2015-12-31. The horizontal axis presents the date and the vertical axis shows the market excess returns.

The graph indicates that daily market excess returns have fluctuated quite widely across time. There were multiple negative drops during long term capital management (LTCM) crisis of 1998, dotcom crash in 2000-2003, financial crisis in 2008-2009, and then European debt crisis in 2011-2012.

In the time series plot of IBM during the same time period, we see even wider fluctuations. In addition to the four crisis in the overall market, 90s were difficult for IBM amid personal computer revolution. IBM is much more risker than the market at least in the first half of the sample.

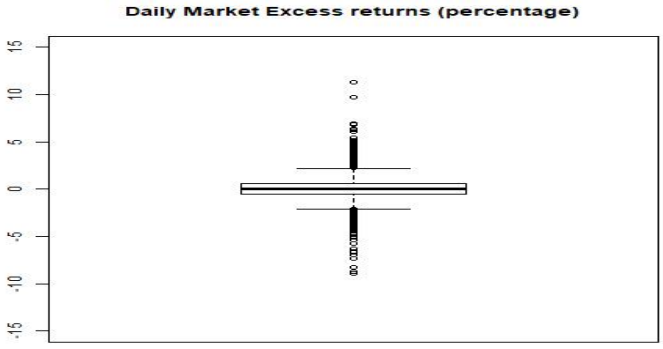
## Step 2: Moments

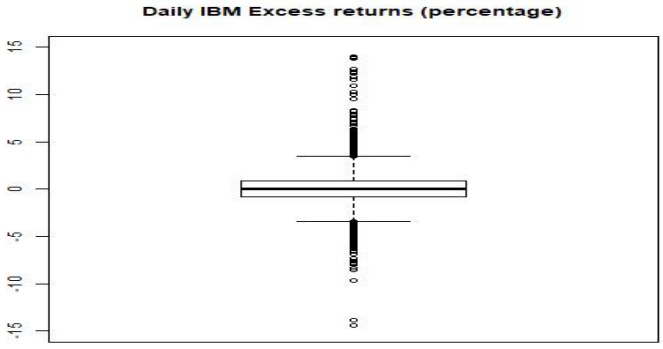
From this step forward, use the daily return series (in percentage unit).

### Step 2.1: (FIG) Box Plot

Create a box plot for market excess returns and another box plot for IBM excess returns.

```
jpeg(filename = "Case1_marketEXERT_Daily.jpeg")
boxplot(marketEXERT, main="Daily Market Excess returns (percentage)",
ylim=c(-15,15))
dev.off()
```



**Box plot**

The box plot (a.k.a. box and whisker diagram) is a standardized way of displaying the distribution of data based on the five number summary: minimum, first quartile, median, third quartile, and maximum. The boxplot is used to map the data to see the spread of all of the different observations. The box shows the middle 50% of the data with the middle horizontal bar as the median and two edges representing first and third quartile.

The whiskers is the flattened arrows extending out of the box. What do they represent and how are they calculated? They represent the reasonable extremes of the data. That is, these are the minimum and maximum values that do not exceed a certain distance from the middle 50% of the data. What is that certain distance? By default in R, it's $1.5 \times IQR$. If no points exceed that distance, then the whiskers are simply the minimum and maximum values.

If there are points beyond that distance, the largest point that does not exceed that distance from above becomes the upper whisker; the smallest point that does not exceed that distance from below becomes the lower whisker. And the points beyond the distance are plotted as single points.

### Step 2.3 (TAB) Numerical Moments: Write Your Own Codes for IBM

For both the daily market excess returns and the daily IBM excess returns, summarize the data by computing the mean (annualized), standard deviation (annualized), skewness, kurtosis, minimum, maximum, Sharpe Ratio (annualized), VaR and expected short fall. In the end, report the correlation between these two returns. Put these statistics in one table.

**Sharpe Ratio**

The Sharpe Ratio is a measure for calculating risk-adjusted return, and this ratio has become the industry standard for such calculations. It was developed by Nobel laureate William F. Sharpe. The Sharpe ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk.

$$SR = \frac{\text{Mean}}{\text{Standard Deviation}} \tag{2.7}$$

How to annualize a Sharpe Ratio?
To annualize the mean, you multiply your mean by 252 because you are assuming the returns are uncorrelated with each other and the return over a year is the sum of the daily returns. You multiply your standard deviation by $\sqrt{252}$ because annualization 252 should be applied to the variance. So the annualization of the ratio is $252/\sqrt{252} = \sqrt{252}$.

```
## Step 2.3: Numerical Moments
install.packages('e1071')
library(e1071)
# Install statistics functions skewness & kurtosis

##Compute Descriptive Statistics for market excess return in daily percentage.
MKTmean<-mean(marketEXERT)*252
MKTsd<-sd(marketEXERT)*sqrt(252)
MKTskew<-skewness(marketEXERT)
MKTkurto<-kurtosis(marketEXERT)
MKTmin<-min(marketEXERT)
MKTmax<-max(marketEXERT)

# Sharpe Ratio
MKTsr<-MKTmean/MKTsd
```

**install.packages(package)**: install packages

- **package**: the name of a package.

**library(package)** load and attach add-on packages.

- **package**: the name of a package.

"e1071" package contains functions for skewness, kurtosis, ...

**skewness(x)**: computes the skewness.

- **x**: a numeric vector containing the values whose kurtosis is to be computed.

**kurtosis(x)**: computes the kurtosis.

- **x**: a numeric vector containing the values whose **excess** kurtosis (kurtosis -3) is to be computed.

**max(x)** return the maximum value of x.

- **x**: numeric or character arguments

**min(x)** return the minimum value of x.

- **x**: numeric or character arguments

```
# Value at Risk
MKTVaR<-quantile(marketEXERT, probs = c(0.05))

#Expected Shortfall
install.packages("PerformanceAnalytics")
library(PerformanceAnalytics)
MKT_raw<-marketEXERT/100
MKTES_raw<-ES(MKT_raw, p=.05,method="historical")
MKTES<-MKTES_raw*100

##Compute Descriptive Statistics for IBM excess return in daily percentage.
... write your own code and name everything such as IBMmean...

## compute the correlation
IBMcMarket<-cor(IBMEXERT, marketEXERT)
```

**VaR and ES**

VaR is the quantile of the return distribution. A 5% VaR means that 95% of the time returns will exceed the VaR and 5% will be worse. The expected shortfall is the conditional expectation of returns given that returns are in the 5% low percentile.

**quantile(x, probs=p)**: computes the sample quantiles q that prob(x $\leq$ q)=p. **x** is a numeric vector and **p** is numeric vector of probabilities with values in [0,1].

```
## Construct each column of our table.
Name<-c("Mean:", "Std:", "Skewness:", "Kurtosis:",
 "Sharpe Ratio","Value at Risk","Expected Shortfall","Correlation:" )
IBM<-c(IBMmean, IBMsd, IBMskew, IBMkurto, IBMsr, IBMVaR, IBMES, IBMcMarket)
Market<-c(MKTmean, MKTsd, MKTskew, MKTkurto, MKTsr,MKTVaR, MKTES, NA)
## Construct table
data.frame(round(IBM,4), round(Market,4),row.names =Name,check.names = TRUE)
}
```

**round(x, digits)** rounds the values in its first argument to the specified number of decimal places

- **x**: a numeric vector.
- **digits**: integer indicating the number of decimal places.

**data.frame(var1, var2, row.names =NULL, check.names = TRUE)** creates data frames, tightly coupled collections of variables which share many of the properties of matrices and of lists, used as the fundamental data structure by most of R's modeling software. The variable names are "var1" and "var2".

- **var1**, **var2**: numeric or character argument
- **row.names** : NULL or a single integer or character string specifying a column to be used as row names, or a character or integer vector giving the row names for the data frame.
- **check.names**: logical. If TRUE then the names of the variables in the data frame are checked to ensure that they are syntactically valid variable names and are not duplicated.

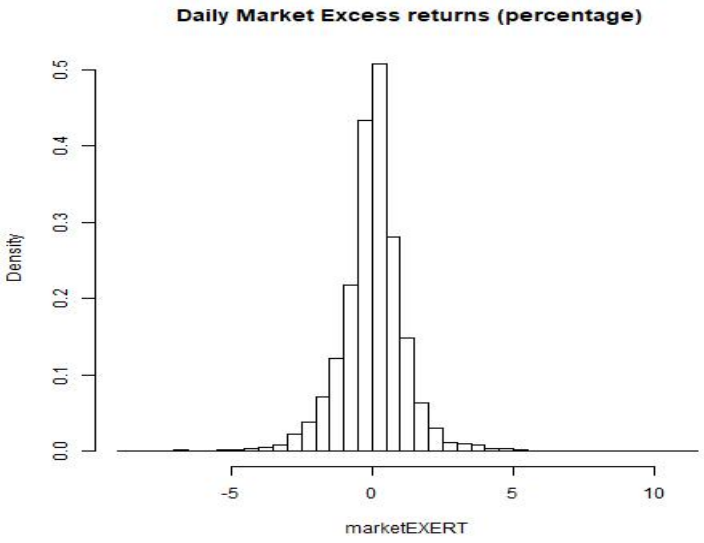
```
                        IBM  Market
Mean:               17.3820  7.8925
Std:                28.8362 18.8361
Skewness:            0.5800 -0.1169
Kurtosis:            8.0602  7.6395
Sharpe Ratio         0.6028  0.4190
Value at Risk       -2.6330 -1.8305
Expected Shortfall  -3.9803 -2.8095
Correlation:         0.5956      NA
```

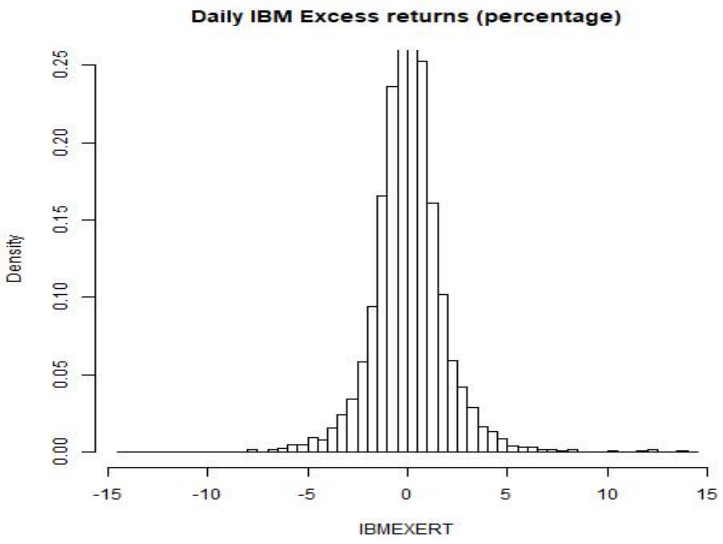
## Step 3: Distribution

### Step 3.1 (FIG) Histogram

Create a histogram plot (probability density function) for market excess returns and another histogram plot for IBM excess returns.

```
jpeg(filename = "Case1_histmarketEXERT.jpeg")
hist(marketEXERT, main = " Daily Market Excess returns (percentage)",
prob=TRUE, breaks = 50)
dev.off()
```



### Histogram plot

Histogram is a graphical method to display a frequency distribution. It is an estimate of the probability distribution of a continuous variable (quantitative variable) and was first introduced by Karl Pearson. On the x-axis we have the values of this variable split into equal-length intervals or so-called bins. On the y-axis we have the frequency of the variable observed in each bin.

Histograms are especially convenient for describing the shape of the data distribution. When data trail off to the right and have a longer right tail, the shape is said to be right-skewed. If data sets with the reverse characteristic  a long, thin tail to the left  are said to be left-skewed. Data sets that show roughly equal trailing off in both directions are called symmetric.
The mode and shape are more obvious from a histogram plot than from a boxplot, while outliers are more obvious from a boxplot.

### hist (x, main =title, xlab = xname, ylab= yname)

hist() computes a histogram of the given data values.

- **x**: a vector of values for which the histogram is desired.
- **main**: the title of the histogram.
- **xlab**: label of the x-axis
- **ylab**: label of the y-axis
- **prob**: if TRUE, the histogram graphic is a representation of probability densities; if FALSE, frequencies.
- **breaks**: a single number giving the number of bins for the histogram

**Hypothesis Test**

**Step 1**: Summarize the statement and the opposite of the statement. Choose the one with "=" sign as the null hypothesis and the other one as the alternative hypothesis.

- $H_0$: Daily excess returns follow a normal distribution
- $H_1$: Daily excess returns don't follow a normal distribution

**Step 2**: Calculate the test statistic and the p-value. Computer outputs usually provide test statistics, degree of freedom and p-value.

**Step 3**: Decision rule: Reject $H_0$ if the probability of observing any value more extreme than the test statistic $< \alpha$. This is because if the null were true, there would be only a small probability (p-value) that we could observe a value more extreme than the test statistic. If we do observe an extreme test statistic value, we don't believe that the null is true, therefore we reject it.

- $\alpha$ is the maximum error you are willing to accept (is given).
- p-value is the measure of evidence against the null hypothesis $H_0$. Smaller p-values indicate more evidence against $H_0$.

**Step 4**: Conclusion. We either reject the null and in favor of the alternative. Or we can't reject the null. We never say we accept the null.

**Step 3.3 (HT) The Jarque-Bera Test** a type of Lagrange multiplier test, is a test for normality. Normality is one of the assumptions for many statistical tests, like the t test or F test; the Jarque-Bera test is usually run before one of these tests to confirm normality. It is usually used for large data sets, because other normality tests are not reliable when n is large (for example, Shapiro-Wilk isnt reliable with n more than 2,000).

Specifically, the test matches the skewness and kurtosis of data to see if it matches a normal distribution. The data could take many forms, including: Time Series Data; Errors in a regression model and Data in a Vector.

A normal distribution has a skew of zero (i.e. its perfectly symmetrical around the mean) and a kurtosis of three; kurtosis tells you how much data is in the tails and gives you an idea about how peaked the distribution is. Its not necessary to know the mean or the standard deviation for the data in order to run the test.

$$JB = n * [s^2/6 + (k-3)^2/24]. \tag{2.8}$$

Where: n is the sample size, $s$ is the sample skewness coefficient, $k$ is the kurtosis coefficient. The null hypothesis for the test is that the data is normally distributed; the alternate hypothesis is that the data does not come from a normal distribution.

```
install.packages('tseries')
library(tseries)
jarque.bera.test(IBMEXERT)
jarque.bera.test(marketEXERT)
```

**jarque.bera.test(x)** tests the null of normality for x using the Jarque-Bera test statistic. This test is a joint statistic using skewness and kurtosis coefficients. Missing values are not allowed.**x** is a numeric vector or time series. A list with class "htest" containing the following components:

- **statistic** the value of the test statistic.
- **parameter** the degrees of freedom.
- **p.value** the p-value of the test.
- **method** a character string indicating what type of test was performed.
- **data.name** a character string giving the name of the data.

**Step 3.4** (HT) **The Lilliefors test** is a test for normality. It is an improvement on the Kolomogorov-Smirnov (K-S) test correcting the K-S for small values at the tails of probability distributions and is therefore sometimes called the K-S D test. Many statistical packages (like SPSS) combine the two tests as a Lilliefors corrected K-S test.

Unlike the K-S test, Lilliefors can be used when you dont know the population mean or standard deviation. Essentially, the Lilliefors test is a K-S test that allows you to estimate these parameters from your sample.

The null hypothesis ($H_0$) for the test is the data comes from a normal distribution. The alternate hypothesis ($H_1$) is that the data doesnt come from a normal distribution. The test assumes that you have a random sample. If the test statistic is significantly large, you can reject the null hypothesis and conclude that the data is not normal.

The Lilliefors test statistic is:

$$D = max|\hat{F(x)} - G(x)| \tag{2.9}$$

where $\hat{F(x)}$ is the empirical cdf of the sample data and $G(x)$ is the cdf of the hypothesized distribution with estimated parameters equal to the sample parameters.

```
install.packages('nortest')
library(nortest)
lillie.test(IBMEXERT)
lillie.test(marketEXERT)
```

**lillie.test(x)** performs the Lilliefors (Kolmogorov-Smirnov) test for the composite hypothesis of normality. **x** is a numeric vector of data values, the number of which must be greater than 4. Missing values are allowed. A list with class htest containing the following components:

- **statistic** the value of the Lilliefors (Kolomogorv-Smirnov) statistic.
- **p.value** the p-value for the test.
- **method** the character string Lilliefors (Kolmogorov-Smirnov) normality test.
- **data.name** a character string giving the name(s) of the data

**Citation**:
1: Jarque, C. M., and A. K. Bera. "A Test for Normality of Observations and Regression Residuals." International Statistical Review. Vol. 55, No. 2, 1987, pp. 163172.
2: Lilliefors, H. W. "On the Kolmogorov−Smirnov test for normality with mean and variance unknown." Journal of the American Statistical Association. Vol. 62, 1967, pp. 399402.